Project Documentation

Project Title: Sustainable Smart City Assistant Using IBM Granite LLM

Team Members:
Team Leader: Vinoth B
Team Member: Tamil Selvan
Team Member: Chandrasekar
Team Member: Mohammed Hashwath Khan

2. Project Overview

Purpose:

The purpose of a Sustainable Smart City Assistant is to empower cities and their residents to thrive in a more eco-conscious and connected urban environment. By leveraging AI and real-time data, the assistant helps optimize essential resources like energy, water, and waste, while also guiding sustainable behaviors among citizens through personalized tips and services. For city officials, it serves as a decision-making partner—offering clear insights, forecasting tools, and summarizations of complex policies to support strategic planning. Ultimately, this assistant bridges technology, governance, and community engagement to foster greener cities that are more efficient, inclusive, and resilient.

Features:
- Conversational Interface: Natural language interaction, allows Q&A; in plain language.
- Policy Summarization: Converts lengthy documents into concise summaries.
- Resource Forecasting: Predictive analytics for energy, water, and waste usage.
- Eco-Tip Generator: Personalized sustainability advice.
- Citizen Feedback Loop: Collects and analyzes public input.
- KPI Forecasting: Projects performance indicators for planning.
- Anomaly Detection: Identifies unusual sensor/usage patterns.
- Multimodal Input Support: Accepts text, PDFs, CSVs.
- Streamlit/Gradio UI: Intuitive dashboard for officials and citizens.

3. Architecture

Frontend (Streamlit): Interactive web UI with dashboards, file uploads, chat, feedback, and reports.
Backend (FastAPI): REST framework powering APIs with async performance and Swagger integration.
LLM Integration (IBM Watsonx Granite): Natural language understanding and generation.
Vector Search (Pinecone): Stores embedded policy docs for semantic search.
ML Modules: Forecasting and anomaly detection using Scikit-learn.

4. Setup Instructions

Prerequisites: Python 3.9+, pip, API keys for IBM Watsonx & Pinecone, Internet.

Installation:

1. Clone repository
2. Install requirements.txt
3. Create .env with credentials
4. Run FastAPI backend
5. Launch Streamlit frontend
6. Upload data & interact with modules

5. Folder Structure

- app/: FastAPI backend logic
- app/api/: API routes
- ui/: Streamlit frontend
- smart_dashboard.py: Main dashboard launcher
- granite_llm.py: LLM integration
- document_embedder.py: Document embeddings with Pinecone
- kpi_file_forecaster.py: Forecasting
- anomaly_file_checker.py: Anomaly detection
- report_generator.py: Report creation

6. Running the Application
- Launch FastAPI server
- Run Streamlit dashboard
- Navigate sidebar
- Upload docs/CSVs
- Chat with assistant, view reports and predictions

7. API Documentation

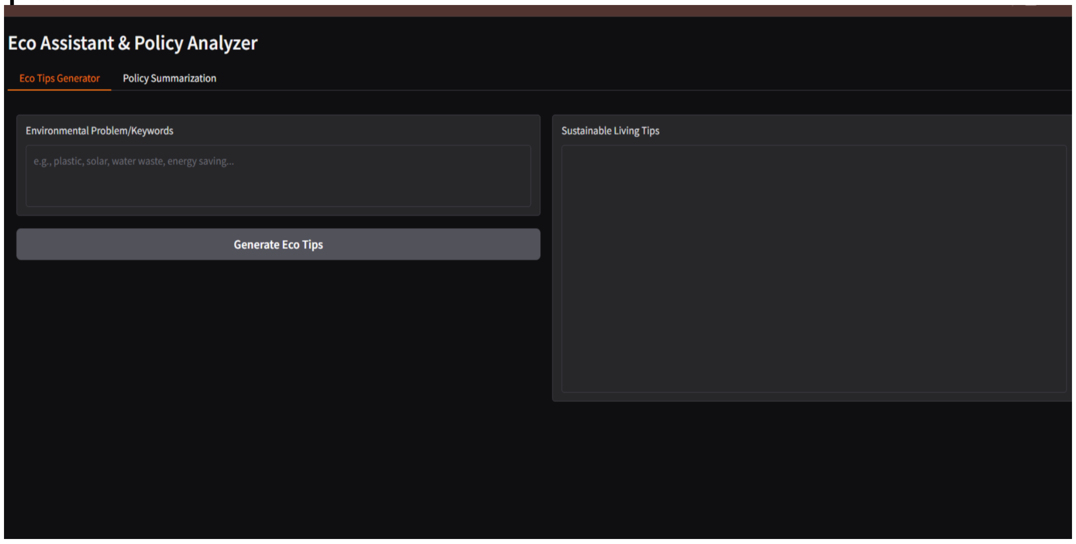
- POST /chat/ask – AI-generated responses
- POST /upload-doc – Upload & embed documents
- GET /search-docs – Semantic search
- GET /get-eco-tips – Sustainability tips
- POST /submit-feedback – Citizen feedback

8. Authentication
- Token-based authentication (JWT, API keys)
- OAuth2 with IBM Cloud
- Role-based access
- Future: sessions & history tracking

9. User Interface

- Sidebar navigation
- KPI visualizations
- Tabs for chat, eco tips, forecasting
- Real-time forms
- PDF downloads

Sample UI Screenshot:


Eco Assistant & Policy Analyzer
Eco Tips Generator
Policy Summarization
Environmental Problem/Keywords
e.g., plastic, solar, water waste, energy saving...
Generate Eco Tips
Sustainable Living Tips

10. Testing

- Unit Testing
- API Testing with Swagger/Postman
- Manual Testing
- Edge Case Handling

12. Known Issues
- Limited scalability in current demo setup
- Requires stable internet connectivity
- No advanced user-role segregation in demo
- Large document handling may cause latency

13. Future Enhancements
- Mobile app integration
- AI-driven traffic & pollution monitoring
- Integration with IoT devices for live data
- Multi-language support
- Advanced analytics dashboards